|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **cmdFile1** | | | | | | |
| **confFile1** | page faults per Run | | | | | |
| 1 | 2 | 3 | 4 | 5 | **AVG.** |
| FIFO | 480 | 480 | 482 | 478 | 481 | **480.2** |
| LRU | 375 | 377 | 378 | 377 | 375 | **376.4** |
| CLOCK | 377 | 377 | 379 | 382 | 377 | **378.4** |

Analysis #7&8

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **cmdFile2** | | | | | | |
| **confFile1** | page faults per Run | | | | | |
| 1 | 2 | 3 | 4 | 5 | **AVG.** |
| FIFO | 11 | 11 | 11 | 11 | 11 | **11** |
| LRU | 9 | 9 | 9 | 9 | 9 | **9** |
| CLOCK | 9 | 9 | 9 | 9 | 9 | **9** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **cmdFile1** | | | | | | |
| **confFile2** | page faults per Run | | | | | |
| 1 | 2 | 3 | 4 | 5 | **AVG.** |
| FIFO | 50 | 50 | 51 | 51 | 49 | **50.2** |
| LRU | 45 | 45 | 47 | 46 | 43 | **45.2** |
| CLOCK | 49 | 47 | 51 | 49 | 48 | **48.8** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **cmdFile2** | | | | | | |
| **confFile2** | page faults per Run | | | | | |
| 1 | 2 | 3 | 4 | 5 | **AVG.** |
| FIFO | 8 | 8 | 8 | 8 | 8 | **8** |
| LRU | 7 | 7 | 7 | 7 | 7 | **7** |
| CLOCK | 7 | 7 | 7 | 7 | 7 | **7** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **cmdFile1** | | | | | | |
| **confFile3** | page faults per Run | | | | | |
| 1 | 2 | 3 | 4 | 5 | **AVG.** |
| FIFO | 45 | 45 | 44 | 46 | 45 | **45** |
| LRU | 42 | 42 | 42 | 43 | 44 | **42.6** |
| CLOCK | 40 | 39 | 40 | 37 | 41 | **39.4** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **cmdFile2** | | | | | | |
| **confFile3** | page faults per Run | | | | | |
| 1 | 2 | 3 | 4 | 5 | **AVG.** |
| FIFO | 6 | 6 | 6 | 6 | 6 | **6** |
| LRU | 5 | 5 | 5 | 5 | 5 | **5** |
| CLOCK | 5 | 5 | 5 | 5 | 5 | **5** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **cmdFile1** | | | | | | |
| **confFile4** | page faults per Run | | | | | |
| 1 | 2 | 3 | 4 | 5 | **AVG.** |
| FIFO | 24 | 18 | 25 | 22 | 25 | **22.8** |
| LRU | 15 | 13 | 15 | 15 | 14 | **14.4** |
| CLOCK | 24 | 17 | 24 | 25 | 17 | **21.4** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **cmdFile2** | | | | | | |
| **confFile4** | page faults per Run | | | | | |
| 1 | 2 | 3 | 4 | 5 | **AVG.** |
| FIFO | 1 | 1 | 1 | 1 | 1 | **1** |
| LRU | 1 | 1 | 1 | 1 | 1 | **1** |
| CLOCK | 1 | 1 | 1 | 1 | 1 | **1** |

# Discussion:

7.) The number of page faults of three algorithms were tested (FIFO, LRU and CLOCK) by entering the same number of pages, the same sequence and the same page frame in main memory. Across all of the above data tables, looking at all the average columns, we observe that FIFO results in most faults, whereas LRU and CLOCK result in significantly lower faults. Run#4 for confFile4 with cmdFile1, was the lone instance where FIFO didn’t result in the worst results. This was simply an anomaly as the CLOCK algorithm made a poor decision in the way it decided to bring in pages into memory leading to the worst result of the three algorithms. FIFO’s logic is based on removing the oldest page in memory and while the implementation for that logic might be easy, the logic itself is rather flawed. This is because a frequently used page is often the oldest, so it will repeatedly be paged out.

Across almost all of the data tables, we saw that LRU resulted in the best results possible, and the CLOCK results were usually just as good, if not almost as good. This falls in line with what we’ve learnt in class that CLOCK’s performance usually falls in between LRU and FIFO. This was as expected as LRU follows the sound principle of locality which allows it to perform nearly as well as the OPT algorithm. However, if we just went by the evidence provided by the table for confFile3 where cmdFile1 is being used, we would be led to believe the CLOCK algorithm is actually better than the LRU. LRU produced an average of 42.6 faults, whereas CLOCK produced only 39.4 faults for this file. This is most likely due to the fact that the commands produced by cmdFile1 in combination with the configuration deployed in confFile3 just happened to be more favorable for CLOCK than LRU. CLOCK functions like FIFO but gives each page a second chance to prove it is useful. For this configuration, not only was it useful, but it was quite ideal.

8.) For this discussion, let’s just focus on tables in which data was collected using only the cmdFile1. It is very easy to see that with increased physical pages allocated to the process, the number of page faults decreases. In fact, just by going from 4 pages in confFile1 to 6 pages in confFile2, we were able to decrease the number of page faults across all algorithms by a factor of 9(on average)! This makes sense, since the more frames we have in physical memory (per process) the less we have to swap pages with disc memory and thus resulting in lower overhead for the system. However, we must keep in mind that increasing the physical memory per process results in lower multiprogramming. This is where we have to be wise in using the working set strategy, where we’re carefully select a resident set resulting in a lower number of page faults.